Appendix A

RVC Toolbox

This appendix present the software tool for modeling and simulating robotic manipulator for analyses under different conditions. This toolbox is a collection of functions written in Matlab and is owned by Prof. Peter I. Corke at Queensland University of Technology and the Director of the ARC Centre of Excellence for Robotic Vision. This toolbox was first developed to help Prof. Corke in his Ph.D. work and later on several revisions and additions take this toolbox into this shape. Presently, tenth edition of the toolbox (RTB 10.3) is available. In the present work, ninth edition of the toolbox RVC 9.1 has been used, released in 2011 with the first edition of the book '*Robotic Vision & Control*'. The first version of this toolbox was released in 1995. This toolbox is more than twenty year old and have gain wide popularity and maturity.

In this toolbox, there are many functions that helpful in several task such as:

- Kinematic Analysis
- Dynamic Analysis
- Trajectory Planning

This toolbox contains both 2D and 3D simulation functions that can represent three dimensional orientation and location by homogeneous transformation. This toolbox also provide functions to convert one data type to another such vector to homogeneous, unit-quaternions. The 3D visual images of the simulation adds better understanding of the programming which also facilitates corrects and feedback of the generated result.

This toolbox also have models of various robots such as PUMA 560, Stanford arm and quadcopter. It also allows the user to build their own model and do kinematic and dynamic analysis. This toolbox supports mobile robotics, path planning algorithms, mapping, localization, and both mapping and localization simultaneously. It also provide Simulink model for non-holonomic vehicle and the Simulink model for a quad rotor flying robot. Various advantages of the toolbox are as follows:

- RVC toolbox provide functions for robot simulation in Matlab which is a higher level language. That allow other tools to be clubed with it.
- It provide faciity to produce your own code based on the built in function.
- It is open source general user license package.

Now, Matlab has developed its own toolbox known as Robotics system toolbox having different functions and tasks, comes with considerable licensing cost. The general user license of the RVC toolbox promotes its large user community. The RVC toolbox is now known as Robotics toolbox (RTB). This toolbox is simple to use and have many demo tutorials for its different functions. Various videos about the toolbox is now available on the internet.

Appendix B

Simulated Annealing

Simulated annealing is a metaheuristic search technique for global minimization of the optimization problem in large search domain. The metallurgical word 'Annealing' is defined as a process of cooling a hot metal to increase the size of its crystals to reduce its defects. The heating and cooling of the material changes its free energy and the temperature which act as variable to manipulate crystal structure. The simulation of the annealing process has been used to find the global minima among the large number domain of optimization problem (Kirkpatrick et al., 1983).

The theory of simulated annealing algorithm lies in the thermodynamic process of rate of change of temperature. In thermodynamic, the energy state of a system is described by the energy state of each of the particles constituting it. The energy state of each particle jumps about randomly, with such transitions governed by the temperature of the system. In particular, the probability $P(\boldsymbol{\varepsilon_i}, \boldsymbol{\varepsilon_j}, \boldsymbol{T})$ of transition from energy $\boldsymbol{\varepsilon_i}$ to $\boldsymbol{\varepsilon_j}$ at temperature T is given by

$$P\left(\boldsymbol{\varepsilon_i} - \boldsymbol{\varepsilon_j}, \boldsymbol{T}\right) = \boldsymbol{e}^{-(\varepsilon_i - \varepsilon_j)/\boldsymbol{T}}$$

From above formula, the probability P of moving from a high-energy state to a lower-energy state is very high. However, there is also a non-zero probability of accepting a transition into a high-energy state, with small energy jumps much more likely than big ones. The higher the temperature, the more likely such energy jumps will occur. Therefore, a physical system, as it cools, seeks to go to a minimum-energy state. For any discrete set of particles, minimizing the total energy is a combinatorial optimization problem. Through

random transitions generated according to the above probability distribution, we can simulate the physics to solve arbitrary combinatorial optimization problems. The steps describing the simulated annealing heuristics are:

Pseudo Code for Simulated Annealing

```
Create initial solution X
Initialize temperature T
repeat
    Iterate for i
        Generate random transition from X to X_i
        If f(X) < f(X_i) then X = X_i
        Else if e((f(X)-f(X_i))/T) > random [0,1) then X = X_i
    Reduce temperature
  Until no change in f(X)
Return X
```

Figure B.1 Simulated annealing algorithm to obtain the global minima

The temperature T is the controlling parameter whose value is kept large in the beginning to involve large number of cost increasing moves and later the temperature is reduced gradually for optimisation. There are some factors on which the successful running of simulated annealing algorithm depends are:

1. **Initial feasible solution :-** Simulated annealing iterates around the initial feasible solution therefore, to obtain the desired global minima initial feasible solution should be properly choosen. Sometimes, if the algorithm doesn't converge to a feasible solution then it is favourable to change the initial feasible solution.
2. **Temperature decrement function :-** It defines the exponential decrement in temperature.

Appendix C

Genetic Algorithm

Genetic algorithm is an evolutionary technique to search the global optimum solution for an objective function. It is a metaheuristic inspired by the process of natural selection involving bio-inspired operators such as crossover, mutation and selection. Genetic algorithm is preferred for high quality solution that demands extensive search in large domain. The primary benefit of this algorithm is that the initial feasible solution is not required. This benefit reduces the initial effort to start the algorithm but indeed increases the search domain that require larger iterations and high computation time. It was developed originally by John Holland in early 1970's.

Genetic algorithm is based on the analogy with the genetic structure which contains chromosomes, their selection, crossover and mutation. The natural selection works on the theory of the survival of the fittest and based on this rule the population of the chromosomes were filtered. Each individual is a possible solution and they have to fight with each other for the resources and mate. The best population is moving forward for competition and another mating. At the end of the process the best individual will survive. The three factors which are affecting the performance of the genetic algorithm:

1. **Selection :-** It is a step in which the better individuals are prefered and forwarded to the next generation. The bett individual is determined by the objective function and optimization problem formed. The portion of the existing population which is allowed to breed a new population is crucial for searching the global optima. In this case the fitness function should be well defined.

2. **Crossover :-** In this process new individual population is created by mating two individual from previous population. It is also called as recombination and by this genetic operator a better next generation have been developed. Genetic algorithm store this information in the form of bit array and crossover between two array can be either single point or two point or uniform.
3. **Mutation** :- By this genetic operator genetic diversity has been maintained from one generation to another. An individual in a population change its original state with a probability. Mutation allows the GA to avoid local minima by changing similar chromosomes.

Pseudo code for Genetic Algorithm
S – set Initialization $t \leftarrow 0$ Initialize P_t to random individuals from S^* Evaluate fitness of individuals **while** (termination condition not meet) do Select individuals from P_t *Recombine and mutate individuals* Evaluate fitness of individuals $P_{t+1} \leftarrow$ new individuals $t \leftarrow t+1$ **return** (set of value of best individuals in P_t)

Figure C.1 Pseudo code for genetic algorithm (Zaritsky and Sipper, 2004)

Limitations:

1. Genetic algorithm has the possibility to converge to a local minima rather than the global minima.
2. It has intangible stopping criterion for every problem and thus gives unreliable and relative optimum solution. It has difficulty in solving complex optimization problems involving large number of variables.